In the Claims:

Please amend claims 1, 8, 10, 11, 12, 13, 14, 15, 16, 17, and 21 as follows:

1. (currently amended)   A method for implementing autonomic testing and verification of software fix programs comprising the steps of:

receiving a software fix program; said software fix program including multiple patches;

sequentially applying each patch of said multiple patches of said software fix program to a software product;

testing said software product responsive to each said sequentially applied patch; ~~and~~

providing test results to a user responsive to said testing of said software product;

sequentially applying iterations of each patch of said multiple patches of said software fix program to a software product and different combinations of said patches to the software product;

calling a test program and receiving expected test results for each applied iteration to the software product;

testing said software product for each applied iteration to the software product; and comparing test results to the expected test results for each applied iteration to the software product; and

saving test results for each applied iteration to the software product and displaying said test results to the user.

2. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 1 wherein said software fix program includes a program temporary fix (PTF); said PTF including multiple patches or programs.

3. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 2 includes the step of providing an isolation manager for receiving said PTF and for sequentially applying each patch or program of said multiple patches or programs of said PTF to said software product.

4. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 3 includes the step of providing a user interface coupled to said isolation manager for receiving user input selections and reporting results to the user.

5. (previously presented)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 2 wherein the step of receiving said software fix program including said program temporary fix (PTF) includes receiving a set of PTFs, each said PTF containing multiple patches or programs; and wherein each patch or program contained in said group of PTFs is sequentially applied to said software product.

6. (previously presented)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 2 includes providing an isolation manager for receiving said software fix program including said program temporary fix

(PTF) and for sequentially applying each patch or program of said multiple patches or programs of said software fix program or said program temporary fix (PTF) to said software product; and wherein said isolation manager, responsive to a manual isolation user option, applies a patch or program of the multiple patches or programs of said software fix program or said program temporary fix (PTF) to said software product, notifies the user of said patch or program applied to said software product, and waits for a user option of next or done.

7. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 6 wherein said isolation manager, responsive to receiving said next user option, applies a next patch or a next program of the multiple programs of said software fix program or said program temporary fix (PTF) to a software product, notifies the user of the next patch or next program applied to the software product, and waits for a user option of next or done.

8. (currently amended)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 2 includes providing an isolation manager for receiving said software fix program including said program temporary fix (PTF) and for sequentially applying each patch or program of said multiple patches or programs of said software fix program or said program temporary fix (PTF) to said software product; and wherein a- said test program and said expected test results are input to the isolation manager; said isolation manager sequentially applies iterations of each patch or program of the multiple patches or programs of said software fix program or said program temporary fix (PTF) and different combinations of the patches or

programs to the software product and calls ~~the~~ said test program for each applied iteration to the software product.

9. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 8 wherein said isolation manager compares test results with the expected test results for each applied iteration and notifies the user when the test results are different from the expected test results.

10. (currently amended)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 2 includes providing an isolation manager for receiving said software fix program including said program temporary fix (PTF) and for sequentially applying each patch or program of said multiple patches or programs of said software fix program or said program temporary fix (PTF) to said software product; and wherein ~~a~~ said test program is input to the isolation manager; said isolation manager sequentially applies iterations of each patch or program of the multiple patches or programs of said software fix program or said program temporary fix (PTF) and different combinations of said patches or programs to the software product and calls the test program for each applied iteration to the software product.

11. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 10 wherein said isolation manager saves test results in a results table for each applied iteration to the software product and displays said test results to the user.

12. (original)   A method for implementing autonomic testing and verification of software fix programs as recited in claim 11 wherein said isolation

manager compares all test results to each other test result to identify a problem patch or program.

13. (currently amended)   Apparatus for implementing autonomic testing and verification of software fix programs ~~including program temporary fixes (PTFs)~~ comprising:

an isolation manager receiving a software fix program ~~PTF~~ containing a plurality of patches ~~programs~~;

a user interface coupled to said isolation manager ~~for~~ receiving user input selections and reporting results to a user; ~~and~~

said isolation manager sequentially applying each patch ~~program~~ of said ~~multiple~~ plurality of patches ~~programs of said PTF~~ to ~~said~~ a software product; and testing said software product responsive to each said sequentially applied program; and providing test results to a user responsive to said testing of said software product; and

said isolation manager sequentially applying iterations of each patch of said multiple patches of said software fix program to the software product and different combinations of said patches to the software product; calling a test program and receiving expected test results for each applied iteration to the software product; testing said software product for each applied iteration to the software product; and comparing test results to the expected test results for each applied iteration to the software product; and saving test results for each applied iteration to the software product and displaying said test results to the user.

14. (currently amended) Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 13 wherein said <u>software fix programs include multiple program temporary fixes (PTFs); each PTF containing a plurality of programs; and wherein said</u> isolation manager is responsive to receiving a set of PTFs, for sequentially applying each program contained in said set of PTFs to said software product, and for testing said software product responsive to each said sequentially applied program.

15. currently amended) Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 13 wherein said <u>software fix programs include multiple program temporary fixes (PTFs); each PTF containing a plurality of programs; and wherein said</u> isolation manager is responsive to a manual isolation user option, for applying a program of the multiple programs of the PTF to said software product, for notifying the user of the program applied to said software product, and for waiting for a user option of next or done.

16. (currently amended) Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 15 wherein said <u>software fix programs include multiple program temporary fixes (PTFs); each PTF containing a plurality of programs; and wherein said</u> isolation manager is responsive to receiving said next user option for applying a next program of the multiple programs of the PTF to said software product, for notifying the user of the next program applied to the software product, and for waiting for a user option of next or done.

17. (currently amended)   Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 13 wherein said software fix programs include multiple program temporary fixes (PTFs); each PTF containing a plurality of programs; and wherein said isolation manager is responsive to receiving a test program; for sequentially applying iterations of each program of the multiple programs of the PTF and different combinations of the programs to the software product and for calling said test program for each applied iteration to the software product.

18. (original)   Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 17 wherein said isolation manager is responsive to receiving expected test results for comparing test results with said expected test results for each applied iteration and for notifying the user when said test results are different from said expected test results.

19. (original)   Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 17 wherein said isolation manager is adapted for saving test results in a results table for each applied iteration to the software product and for displaying said test results to the user.

20. (previously presented)   Apparatus for implementing autonomic testing and verification of software fix programs as recited in claim 19 wherein said isolation manager is adapted for comparing all test results to each other test result to identify a problem program.

21. (currently amended)   A computer program product for implementing autonomic testing and verification of software fix programs ~~or program temporary fixes (PTFs)~~ in a computer system, said computer program product including a plurality of computer executable instructions stored on a computer recording medium, wherein said instructions, when executed by the computer system to cause the computer system to perform the steps of:

receiving a <u>software fix</u> program<u>,</u> ~~temporary fix (PTF); said PTF~~ <u>said software fix program</u> including multiple <u>patches</u> ~~programs~~;

sequentially applying each <u>patch</u> ~~program~~ of said multiple <u>patches</u> ~~programs~~ of said <u>software fix program</u> ~~PTF~~ to a software product;

testing said software product responsive to each said sequentially applied <u>patch</u> ~~program~~; ~~and~~

providing test results to a user responsive to said testing of said software product<u>;</u>

<u>sequentially applying iterations of each patch of said multiple patches of said software fix program to a software product and different combinations of said patches to the software product;</u>

<u>calling a test program and receiving expected test results for each applied iteration to the software product;</u>

<u>testing said software product for each applied iteration to the software product; and comparing test results to the expected test results for each applied iteration to the software product; and</u>

saving test results for each applied iteration to the software product and displaying said test results to the user.